REMARKS

Introduction

Claims 1, 3-11 and 13-21 are pending in the application, of which claims 1, 11 and 21 are in independent form. Claims 1, 3-6, 8, 10, 11, 13-16, 18, and 19-21 have been amended by this Amendment.

Rejections under 35 U.S.C. § 101

Claims 1, 3-6, 10-11, 13-16, and 20-21 stand rejected under 35 U.S.C. § 101, as allegedly being directed to non-statutory subject matter. By this Amendment, the pending claims have been amended to address the 35 U.S.C. § 101 rejections. More specifically, independent claims 1, 11, and 21 have been amended so that the absolute total number of software defects for the upcoming software are displayed on a display device to a user. Claims dependent therefrom i.e., claims 3-6, 10, 13-16, and 20, now also include this limitation. Support for a display device is found in the application as published in paragraph [0017]. Accordingly, withdrawal of the rejections to 1, 3-6, 10-11, 13-16, and 20-21 under 35 U.S.C. § 101, is requested.

Rejections under 35 U.S.C. § 103(a)

Claims 1, 3, 6, 11, 13, 16 and 21 stand rejected under 35 U.S.C. § 103(a) as being anticipated by "Gauging Software Readiness with Defect Tracking," 1997 IEEE Software, pp. 135-136 (*McConnell*) in view of N. E. Fenton, et al., "A Critique of Software Defect Prediction Models," IEEE Transactions on Software Engineering, 1999, pp. 675-689 (*Fenton*).

The Examiner cited primarily paragraphs 4 and 16 of *McConnell* to describe the methods of predicting defects in a future release. Paragraph 4 of *McConnell* describes a method for predicting defects of a future release based on "defect density" (see paragraph 2 of

McConnell) in which defect tracking relies upon "the number of defects per line of code." (*McConnell* at ¶ 1)(underline added). *McConnell* also describes measuring "defects per 1,000 lines of code (KLOC)." (*Id.* at ¶ 2)(underline added). In paragraph 16, *McConnell* describes a method for predicting defects of a future release based on a combination of approaches. These approaches include the "defect density" method of paragraph 2 and 4, the "defect pooling" method of paragraphs 6-8, and the "defect seeding" method of paragraphs 9-12. In the "defect pooling" method, the number of future defects is predicted based on dividing the number of defects actually found into two arbitrary pools, and then predicting the number of future defects based on the number of defects in one pool times the number of defects in the second pool divided by the number of defects in the sum of the pools. In the "defect seeding" method, a number of artificial or "seeded" defects is deliberately inserted into the program to be tested. The number of future "unseeded" or "indigenous" defects is predicted to be the number of seeded defects planted divided by the number of seeded defects found times the number of indigenous defects found.

Amended claim 1 recites, *inter alia*, "[a] method for predicting the number of software defects for an upcoming software release includes the step of ... forecasting the absolute total number of software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release ... wherein determining the relative size of the upcoming software release includes the steps of: determining the number of new test requirements for the upcoming software release; determining the number of test requirements for the baseline software release; and dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release."

McConnell does not teach, suggest, or provide motivation for all of the features recited by amended claim 1 of the present application. For example, *McConnell* does not describe "forecasting the absolute total number of software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release" and "determining the number of new test requirements for the upcoming software release; determining the number of test requirements for the baseline software release; and dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release." In stark contrast, *McConnell* merely describes measuring defects per line of code or per 1,000 lines of code (KLOC) for the "defect density" method. The "defect pooling" method divides current defect reports into two arbitrary groups, not absolute total number of defects. The "defect seeding" method bases future defect prediction on artificially seeded defects, not indigenous (real) absolute total defects. The methods described by *McConnell* of predicting defects is very different than the claimed use of "absolute total number of indigenous defects" and "test requirements," as recited by amended claim 1 of the present application.

Accordingly for at least these reason, applicant submits that amended claim 1 is allowable over *McConnell.*

Fenton fails to correct the deficiencies of *McConnell. Fenton* describes a method for predicting the number of defects in a future release based on "Bayesian Belief Networks." Before delving into "Bayesian Belief Networks," *Fenton* describes other prediction methods. One method predicts future defects based on size and complexity metrics. These metrics base future defect prediction on the sum of the defects found during testing and the defects found during two months after release and also uses defects per lines of code similar to *McConnell.*

Other size approaches involve the volume of unique operators and operands in the code. *Fenton* describes techniques based on testing metrics, in which future defects are predicted based on collecting data from early inspection and testing phases, including the number of paths covered in the code. *Fenton* also describes techniques based on process quality data, or predicting the quality future code based on the quality of the development process. The primary approach advocated by *Fenton,* i.e. "Bayesian Belief Networks," involves creating a graphical network that represents the probabilistic relationships among variables. The graphical networks are graphs made up of nodes and arcs, wherein the nodes represent uncertain variables and the arcs represent the causal/relevance relationship between variables. The methods described by *Fenton* of predicting defects is very different than the claimed use of "absolute total number of indigenous defects" and "test requirements," as recited by amended claim 1 of the present application.

Claims 11 and 21, while different in scope than claim 1, recite features similar to those discussed above with respect to claim 1. For example, claim 11 recites, *inter alia*, "a processor for determining the relative size of the upcoming software release with respect to a baseline software release and forecasting the absolute total number of indigenous software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release" and "wherein the information obtained by the input device includes the number of new test requirements for the upcoming software release and the number of test requirements for the baseline software release, and the processor determines the relative size of the upcoming software release by dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release" while claim 21 recites, *inter alia*, "forecasting the absolute total number of indigenous software defects for the

upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release;" and "wherein determining the relative size of the upcoming software release includes the steps of: determining the number of new test requirements for the upcoming software release; determining the number of test requirements for the baseline software release; and dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release."

Accordingly, each of independent claims 11 and 21 are deemed to be allowable, at least for reasons discussed above with respect to the allowability of claim 1.

Each of claims 3, 6, 13, and 16 depend from one of claims 1 and 11, that have been discussed above, and are believed to be allowable, and further narrow and define those claims. Therefore, at least for these reasons, claims 3, 6, 13, and 16 are also believed to be allowable.

Claims 4, 5, 7, 8, 14, 15, 17 and 18 stand rejected under *McConnell* in view of *Fenton* and further in view of "An Analysis of Several Software Defect Models," IEEE Transactions on Software Engineering, vol. 14, no. 9, September 1988 (*Yu*).

As described above, neither *McConnell* nor *Fenton* describes, teaches, or provides motivation for all of the features recited by claims 1 and 11.

Yu does not cure the deficiencies of *McConnell* and *Fenton.*

Yu describes (and is cited in the office action for the purpose of describing) several software defect models that incorporate regression and re-factoring factors.

Yu, either alone, or in a hypothetical combination with *McConnell* and/or *Fenton*, does not describe, teach, or provide motivation for a system or method includes the step of "forecasting the absolute total number of indigenous software defects for the upcoming software

release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release," and "wherein determining the relative size of the upcoming software release includes the steps of: determining the number of new test requirements for the upcoming software release; determining the number of test requirements for the baseline software release; and dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release," as recited by claim 1, or "a processor for determining the relative size of the upcoming software release with respect to a baseline software release and forecasting the absolute total number of indigenous software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release" "wherein the information obtained by the input device includes the number of new test requirements for the upcoming software release and the number of test requirements for the baseline software release, and the processor determines the relative size of the upcoming software release by dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release," as recited by claim 11.

For at least these reasons, claims 1 and 11 are deemed to distinguish patentably over any hypothetical *McConnell-Fenton-Yu* combination.

Each of claims 4, 5, 7, 8, 14, 15, 17 and 18 depend from one of claims 1 and 11, that have been discussed above, and are believed to be allowable, and further narrow and define those claims. Therefore, at least for these reasons, claims 4, 5, 7, 8, 14, 15, 17 and 18 are also believed to be allowable over any hypothetical *McConnell-Fenton-Yu* combination.

Claims 9, 10, 19, and 20 stand rejected under *McConnell* in view of *Fenton* and further in view of U. S. Patent No. 6,477,471 (*Hedstrom*).

As described above, neither *McConnell* nor *Fenton* teaches, suggests, or provides motivation for all of the features recited by claims 1 and 11.

Hedstrom does not cure the deficiencies of *McConnell* and *Fenton*. *Hedstrom* describes a method for predicting defects based on historical data of defects at different stages of development and a value representing a goal for escaping defects. The goal for the number of escaping defects and planned number of opportunities for defects are back-solved to determine the total number of defects. The total number of defects are distributed as a function of the historical data to provide prediction of defects at the different stages of development. *Hedstrom* uses as input the number of software lines of code similar to *McConnell*. At a given stage of development, there a certain number of defects which have a Poisson distribution. Future defects are calculated from a complex set of summations including the Poisson distribution. The methods described by *Hedstrom* of predicting defects is very different than the claimed use of "absolute total number of indigenous defects" and "test requirements," as recited by amended claims 1 and 11 of the present application.

Hedstrom, either alone, or in a hypothetical combination with *McConnell* and/or *Fenton*, does not describe, teach, or provide motivation for a system or method includes the step of "forecasting the absolute total number of indigenous software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release;" and "wherein determining the relative size of the upcoming software release includes the steps of: determining the number of new test requirements for the upcoming software release; determining the number of test requirements for the baseline software release; and dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release," as recited by claim 1, or "a processor for determining the relative

size of the upcoming software release with respect to a baseline software release and forecasting the absolute total number of indigenous software defects for the upcoming software release based on the relative size of the upcoming software release and the absolute total number of observed indigenous software defects for the baseline software release" "wherein the information obtained by the input device includes the number of new test requirements for the upcoming software release and the number of test requirements for the baseline software release, and the processor determines the relative size of the upcoming software release by dividing the number of new test requirements for the upcoming software release by the number of test requirements for the baseline software release" as recited by claim 11.

For at least these reasons, claims 1 and 11 are deemed to distinguish patentably over any hypothetical *McConnell-Fenton-Hedstrom* combination.

Each of claims 9, 10, 19, and 20 depend from one of claims 1 and 11, that have been discussed above, and are believed to be allowable, and further narrow and define those claims. Therefore, at least for these reasons, claims 9, 10, 19, and 20 are also believed to be allowable over any hypothetical *McConnell- Fenton- Hedstrom* combination.

Thus, applicant submit that each of the claims of the present application are patentable over each of the references of record, either taken alone, or in any proposed hypothetical combination. Accordingly, withdrawal of the rejections to the claims is respectfully requested.

Conclusion

In view of the above remarks, reconsideration and allowance of the present application is respectfully requested. If any additional fee is due, please charge the required fee to deposit account number 50-1358. Applicant's undersigned agent may be reached by telephone at (973) 597-2500. All correspondence should be directed to our address listed below.

Respectfully submitted,

Date: 5/1/07

Raymond G. Cappo
Patent Agent for Applicant
Registration No. 53,836

Docket Administrator
Lowenstein Sandler PC
65 Livingston Avenue
Roseland, NJ 07068